Ensemble methods

- Combine results of multiple classifiers
  - Different learners
  - Different training data subsets
  - Combined predictions
    - averaging
    - weighted voting
    - model of combination
- Helps tackling error components
  - Bias
    - Model assumptions (e.g., linear separation, shallow decision tree)
    - Consistently incorrect for some instances
  - Variance
    - Data variations leading to very different models (~ overfitting)
    - Inconsistently classified data
  - Noise
    - Intrinsic error in target class
    - Some algorithms are more affected, some less

Bagging

- Single learning algorithm
- *k* data samples with replacement
- *k* learned (same kind) classifiers
- Majority vote
- Reduces variance (makes sense for low bias learners, e.g. deep trees)
- Models are independent, can be built in parallel

Boosting

- Single learning algorithm
- Weighted training instances
  - Adapted learning
  - Weighted data sampling
- Iterative reweighting according to classifier performance
- Focus on misclassified instances in next iteration (increased weights)
  - Various increase approaches and termination criteria
- Aggregation of weighted (according to performance) predictions
- Reduces overall bias (to be used with simple, high bias, models)
- Reduction of variance depends on intensity of reweighting scheme
  - No reweighting in iterations == bagging
- Sensitive to noise (training can focus on bad data!)
- Models depend on previous ones, sequential process

Stacking

- Combination of predictions with another machine learned model
- Two level classification, two data subsets
- *k* classifiers (bagged, boosted or from different learners) learned on the first subset
- Their outputs on second subset are *k* new features
- Second level classifier is trained on
  - new feature space (of size *k*), or
  - combined feature space (old+new)

Random forest

- Similar as bagging with decision trees, but promotes more diverse trees
  - Decision trees in bagging tend to be similar
- Randomness at splits:
  - A random subset of attributes
  - Often advised: $\log_2(\#all\_atribs)+1$
- Usually no or minimal pruning
- Also bootstrapped data samples (as in bagging)
- Majority vote
- Efficient (less attributes considered at splits)
- Resistant to noise, outliers and overfitting
- Not useful in case of just a few features
- Can perform badly in case of a small percentage of relevant features

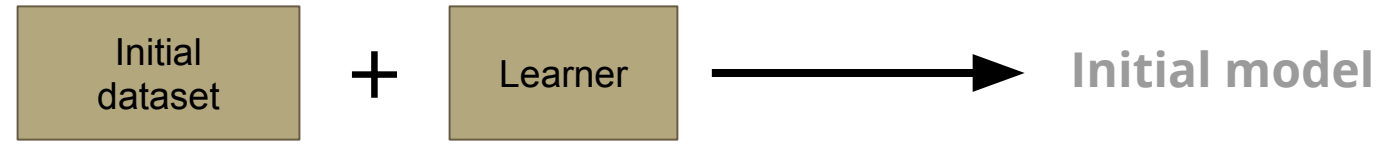
Gradient boosting

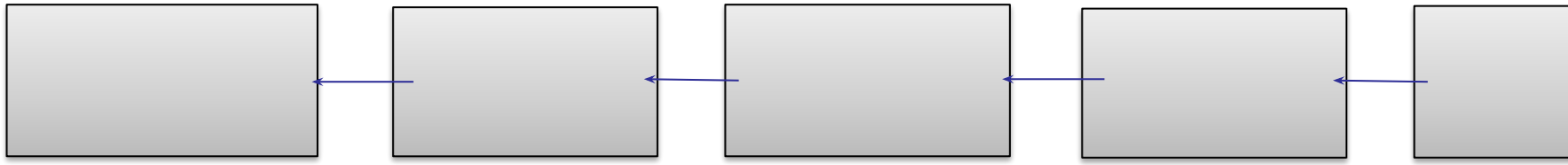
- Idea: construct a (regression tree) model and let the residuals (y - y') become a new target for another model in the iteration
- Next model learns the residuals of the first one
  - Using original features and the new target
- Adding the two models together we get a better prediction
- Repeat until stopping criterion...
- In fact: a gradient of a (any differentiable) loss function is usually modelled instead of residuals (added parameter: step size)
- Sensitive to noise
- Sequential (cannot run in parallel)

Active learning

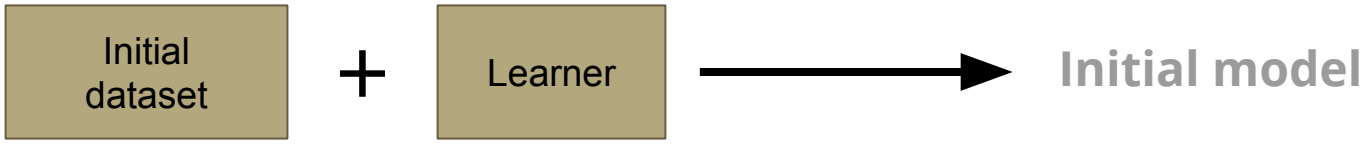
- Labels are sometimes hard or expensive to get
  - Time restrictions in dynamic settings
- AL aims at getting the most of information with the least amount of labels
  - Integration of labeling and classification
  - It is not assumed (as usually) that training data is labeled and other data is not
- Components
  - Querying system : selects the instances to be labelled
  - Oracle : provides labels (human, costly system)
- Stopping conditions
  - exhausted budget
  - no further improvements of performance
  - can be used continuously
    - e.g., to address concept drift in a data stream setting

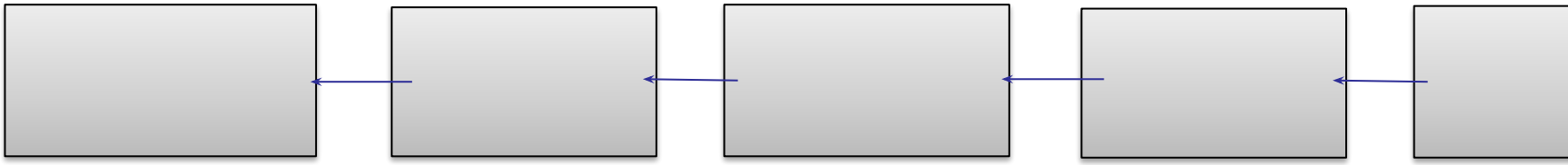
Active learning

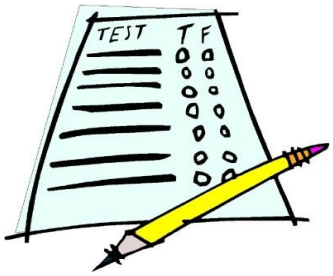




Active learning

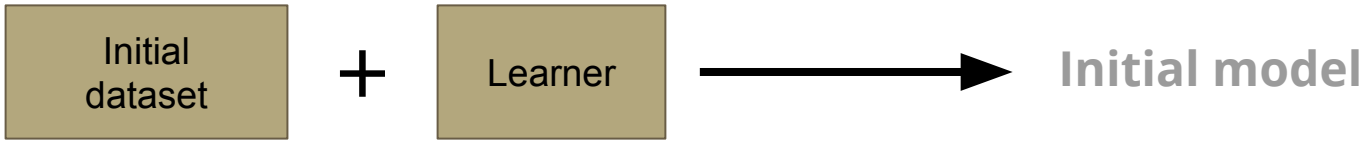


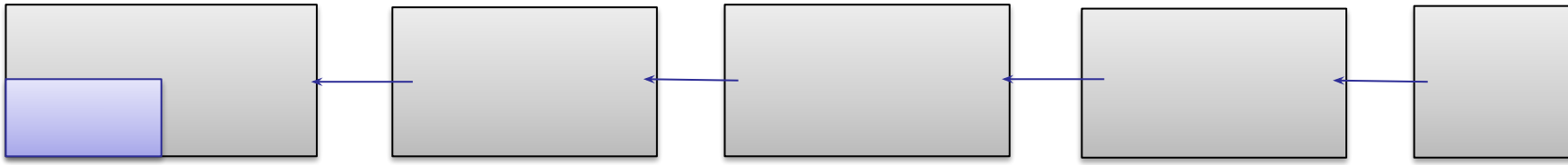


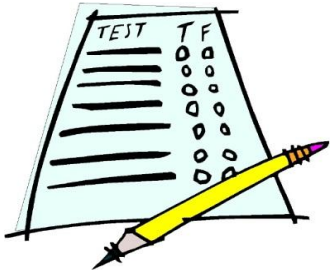


Initial model

Active learning

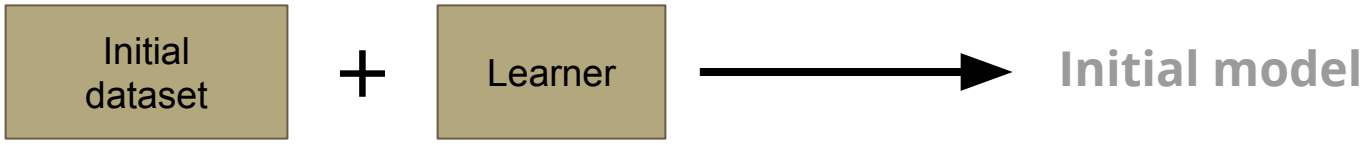


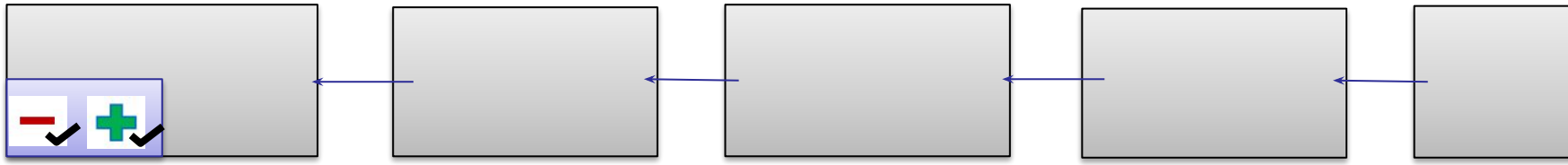


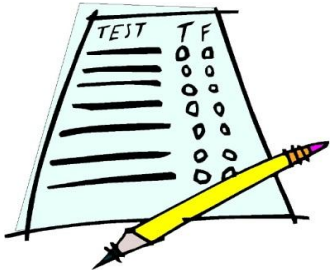


Initial model

Active learning

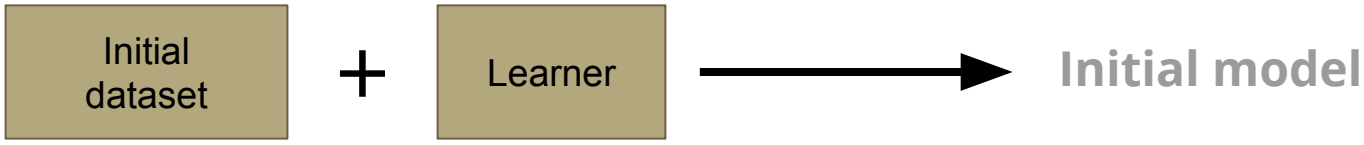


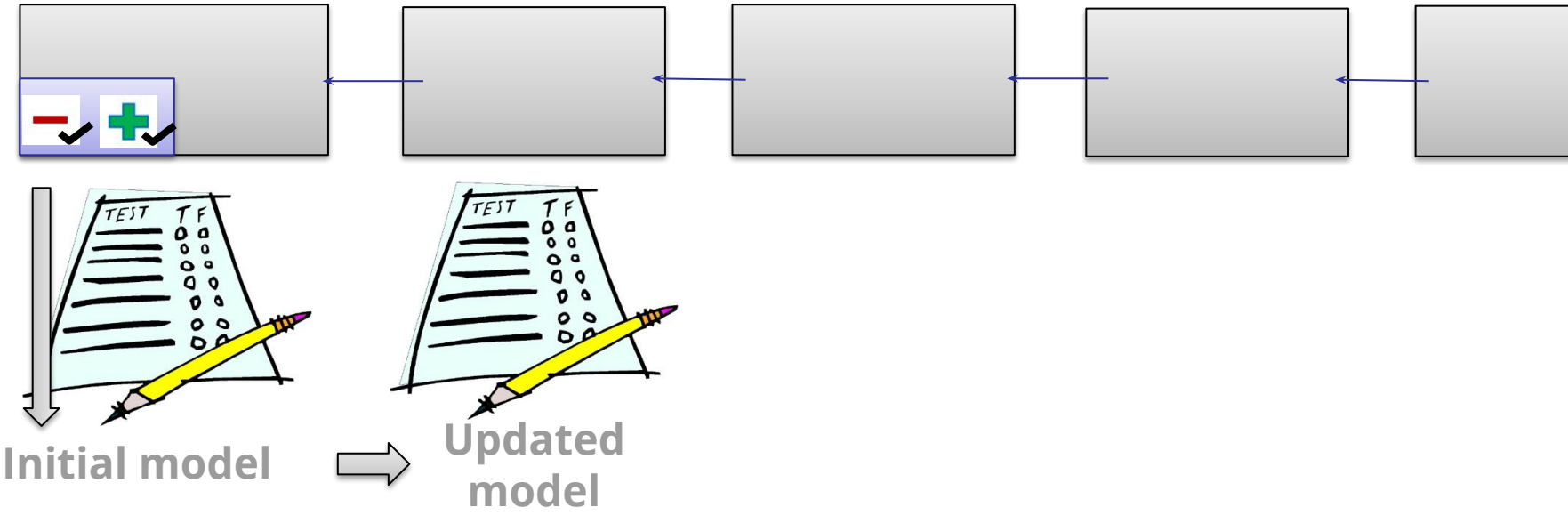




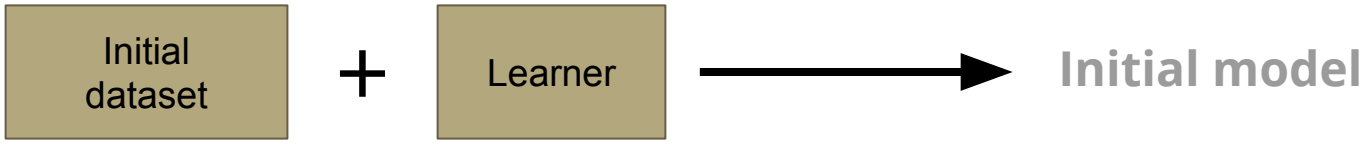
Initial model

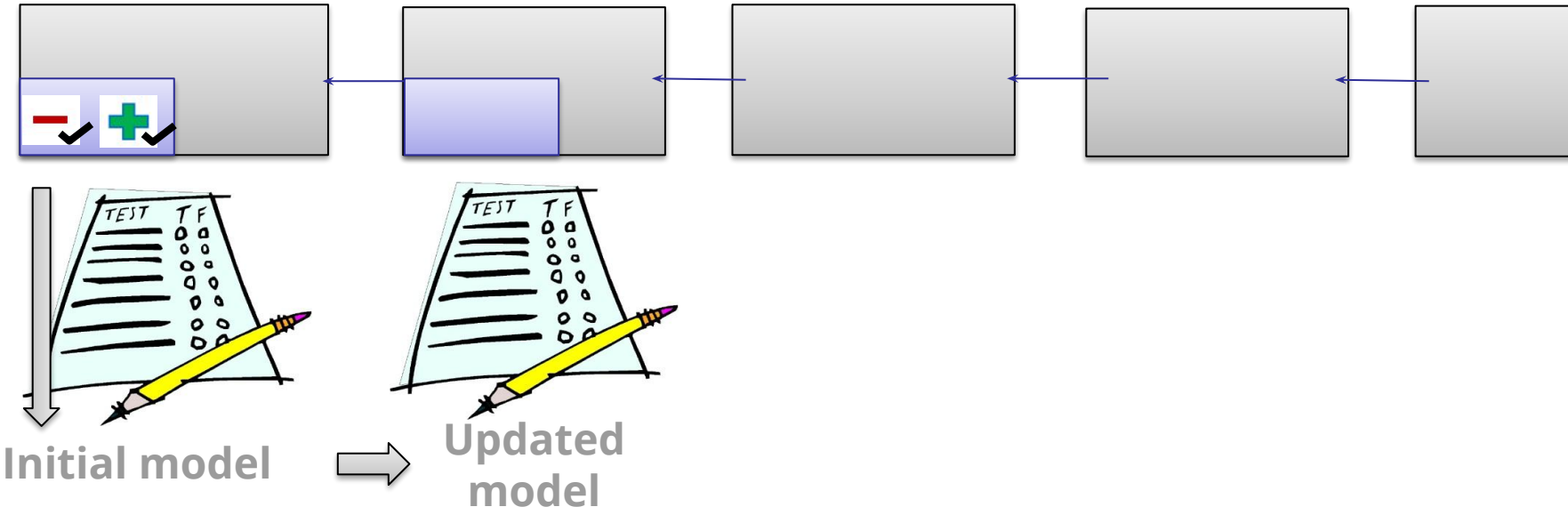
Active learning



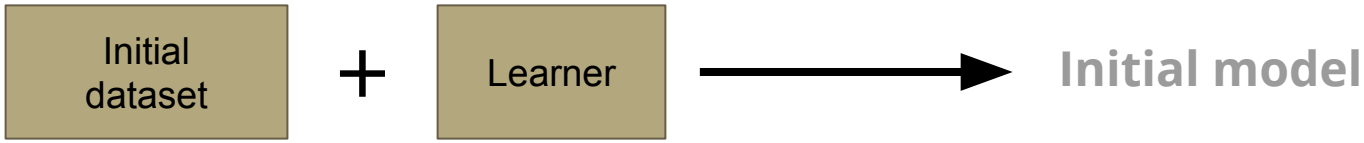


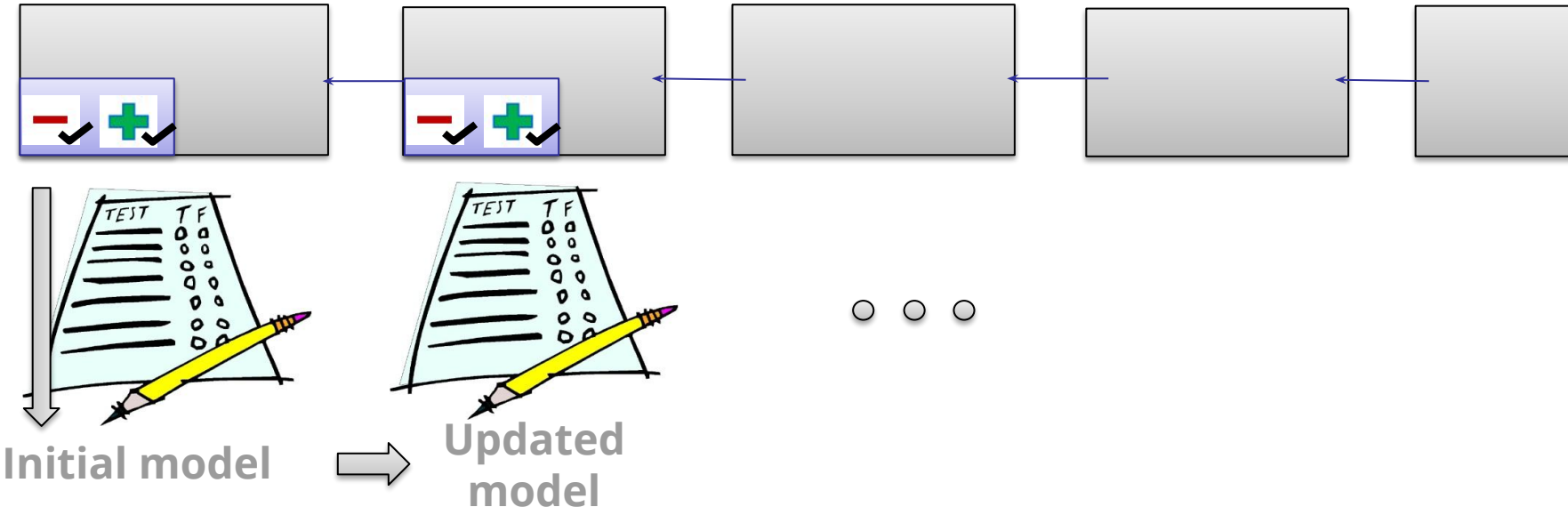
Active learning





Active learning





Active learning - querying strategies

- Querying strategies
  - Highest uncertainty regions
    - danger: querying in low data quality areas, unrepresentative outliers
  - Performance-based
    - pre-quantify impact of adding an instance on the rest of the unlabeled
    - e.g., which one would increase the confidence of the other classifications the most?
    - + : entire dataset in focus, not only the queried instance
  - Representativeness
    - equal representativeness (matching density distribution of data)
    - e.g., average similarity to all other instances
  - And many others and their mixtures...
    - a common one: highest uncertainty + density weighted random

Practical issues and considerations

- Test data leaks
  - e.g., proper oversampling
- Label leaks (disguised)
  - Return, ..., Market adjusted return, ...
  - Member_id, ..., on_site_activity, purchase_made, ..., Yt2/Yt1 payments ratio, ...
- Benchmarking
  - Importance of including dummy classifiers
    - majority class
    - average
- Do not neglect comprehension
  - Models sometimes “work” by chance
    - Numerous hypotheses testing
    - Bonferroni correction for significance levels (p/#test)
  - Learners sometimes model something else (adversarial examples)
- Most importantly : know the problem and check your data!